University of Colombo, Sri Lanka

University of Colombo School of Computing

DEGREE OF BACHELOR OF INFORMATION TECHNOLOGY (EXTERNAL)

Fourth Semester Examination - 2025

IT4406 — Agile Software Development

One (1) Hour

Answer All Questions

To be completed by the candidate								
Index Number								

Important Instructions to candidates:

- Please ensure that you have the correct examination paper in front of you.
- Students should answer in the medium of English language only using the space provided in this question paper.
- Note that questions appear on both sides of the paper. If a page or a part of this question paper is not printed, please inform the supervisor immediately.
- Clearly write your index number on each page of this Question paper.
- This paper consists of **2** questions in **8** pages (including the Cover Page).
- The duration of the paper is **One** Hour.
- Answer **ALL** questions.
- Do not tear off any part of this answer book. Under no circumstances may this book (or any part of this book), used or unused, be removed from the Examination Hall by a candidate.
- Programmable or Non-Programmable Calculators and any electronic device capable of storing and retrieving text, including electronic dictionaries, smart watches and mobile phones, are not allowed.

To be completed by the examiners

1	
2	
Total	

Index Number | | | | | | | | |

1. A development team notices that at the end of each iteration, a significant number of bugs and integration issues emerge. Testing activities are typically postponed until the final stages of development, and code contributions from different team members frequently result in integration failures.

(a). List down **three** specific practices or processes based on (or influenced by) Extreme Programming (XP) that could help address these issues.

[3x2: 6 marks]

[Question 1 is based on 1. Introduction to Agile Software Development, 8. Introduction to Testing in SCRUM (e.g., TDD, CI), 9. Extreme Programming (XP) sections of the syllabus. The focus is on the *application* of concepts from these sections for solving the noted problems.]

- Test-Driven Development (TDD) – Writing tests before code implementation so that development is guided by testing criteria.
- Continuous Integration (CI) – Frequently merging code changes into a shared repository and running automated tests on each integration.
- Pair Programming – Two developers working together at one workstation to co-author code in real time.

Naming the practice and/or a brief explanation of the practice or process is sufficient.

(b). For **each** recommended practice or process in (1a), explain how it helps address the noted problems. Include concrete examples and expected outcomes.

[3x4: 12 marks]

- TDD: By writing unit tests early and continuously, developers get immediate feedback on whether new code meets requirements. This practice ensures high unit-test coverage and catches bugs at the moment of coding (not weeks later). For example, if a function is developed with TDD, any logic error is caught by a failing test right away, preventing that bug from ever reaching integration.

- CI: CI addresses integration issues by merging code from all team members frequently (at least daily or for every commit) and running an automated build and test suite each time. This means integration conflicts or incompatible changes are identified immediately when a change is introduced.

Index Number | | | | | | | | |

- Pair Programming: Having two programmers collaborate on the same code simultaneously means real-time code review and knowledge sharing. Many mistakes or bad design choices are caught as the code is being written, rather than during a later integration or testing phase.

The answers to this question will be based on your answers in 1a.

(c). Identify and explain **three** indicators or metrics a team could use to assess whether newly introduced XP processes or practices are addressing problems such as frequent bugs and poor integration. **Note**: For this question, you can use your answers in (1a) or choose any other XP process or practice.

[3x3: 9 marks]

- Defect density: Track the number of bugs found at the end of each iteration (or the number of production defects reported). A downward trend in bugs reported (or a lower defect density in the code) after adopting CI/TDD/pair programming would indicate improved code quality.
- Build success rate: Measure the frequency of broken builds or failed integrations. A healthy trend would be that build failures become rare and the integration success rate goes up after adopting the XP processes or practices.
- Lead time for changes: Measure the time from code commit to successful deployment or integration. A reduction after Agile/XP adoption indicates faster and smoother integration.

Index Number | | | | | | | | |

(d). Identify and explain **two** challenges a team might face when transitioning from a traditional (e.g., waterfall) based environment to working with XP practices or processes.

[2x4: 8 marks]

- Cultural and Mindset Resistance: Team members may resist the change in mindset that Agile/XP requires. In a waterfall environment, people are used to upfront planning, separate tester/developer roles, and individual code ownership. Switching to Agile/XP means embracing collective code ownership, constant collaboration (pairing), and incremental planning. Some members might find it uncomfortable at first.

- Skill and Process Adaptation Challenges: Adopting practices like TDD and CI can initially slow the team down if they lack the knowledge. Teams coming from a waterfall setting may also lack automated testing skills or continuous integration tools/infrastructure.

(e). Consider the paradigm shift proposed in the Agile Manifesto. Using these principles or values, explain how the noted problems in (1d) can be addressed using **three** specific examples.

[3x5: 15 marks]

- *Individuals and interactions over processes and tools*
  To reduce cultural resistance, emphasise teamwork and communication as values. Example: Instead of imposing pair programming as a rigid process, hold a team retrospective to discuss it openly (a practice aligning with *individuals and interactions*). Let developers share their concerns and demonstrate the benefits in a friendly way. Pair junior team members with more experienced agile-focused members.

- *Working software over comprehensive documentation*
  To ease *skill and process adaptation challenges*, show how practices like TDD and CI help maintain working software continuously. Instead of requiring extensive design specs, encourage writing small, testable increments of code. This reduces pressure and helps teams gradually become proficient in automated testing, as they get fast feedback and build confidence with each successful test run.

Index Number | | | | | | | | |

- *Responding to change over following a plan*
  Both challenges (cultural resistance and skill gaps) are intensified by rigid plans. Emphasising adaptability allows the team to incrementally adopt XP practices. For example, the team might start with simple CI scripts and basic unit tests rather than a full CI/CD pipeline from day one. Adopting practices iteratively based on feedback reduces fear and failure risk.

2. A Scrum team is working on a high-priority project that spans multiple sprints. The team is preparing for its next release and wants to ensure that its planning and sprint execution align with Agile values.

(a). Identify **three** core planning principles in Scrum that contrast with traditional project planning. Briefly explain how **each** principle can help the team described above reduce waste or increase responsiveness.

[3 × 3 = 9 marks]

This question is based on 6. Scrum Planning and 7. Sprints sections of the syllabus.

- *Defer decisions until the last responsible moment.*
  Hold off on making detailed decisions until you actually have the best possible information. This means that you only detail the requirements when they are about to be worked on. This greatly reduces waste as you're not documenting or analysing features that might be cut or changed later. It also increases responsiveness by keeping options open. Here, the team can incorporate late-breaking changes or new ideas without the pain of undoing months of plans.

- *Favour smaller, more frequent releases.*
  Facilitates quicker feedback and adaptability, reducing risk from large batch releases. Compared to a traditional plan with a single delivery, small releases reduce waste by shortening the feedback cycle. The team and stakeholders can inspect a working product increment early and often. This prevents building large batches of features that turn out to be misaligned with user needs.

- *Plan to learn fast and pivot when necessary or Plan adaptively, not predictively.*.
  This encourages experimentation and adapting plans based on insights from working increments, resulting in increased responsiveness.

Index Number | | | | | | | | |

(b). The team is planning its next release. The Product Owner suggests using a fixed-date release approach.

i. Outline **four** key steps involved in performing fixed-date release planning.

[12 marks]

- Determine the number of sprints for the release.
- Groom backlog and estimate items.
- Measure or estimate the team's velocity. (range).
- Multiply velocity by the number of sprints to estimate feature delivery.

ii. How does a fixed-date release approach help the team forecast *"will-have"* and *"might-have"* features?

[6 marks]

- A fixed-date release plan provides a transparent method to forecast scope under a non-negotiable timeline by using the team's known throughput. By fixing the date and varying scope, the team uses its velocity range to set expectations about *what will definitely be done* versus *what might be done* by that date. This helps stakeholders set realistic expectations.

Index Number | | | | | | | | |

(c). The company also wants to ensure that the overall product strategy is aligned with business value delivery.

i. What is the purpose of Portfolio Planning in Scrum?

[3 marks]

- To prioritise and align development across multiple products with strategic business goals.
- In practice, the purpose is to decide which initiatives (products, projects) to start or continue, in what order, and for how long, so that the company's limited resources yield the maximum business benefit.

ii. Name and describe **two** inflow or outflow strategies (one each) used in Portfolio Planning.

[8 marks]

- *Inflow – Apply Economic Filter:* Prioritize high-value items and eliminate low ROI initiatives. [Other answers: Balance the Arrival Rate with the Departure Rate. Quickly Embrace Emergent Opportunities. Plan for Smaller, More Frequent Releases]
- *Outflow – Establish WIP (Work in Progress) Limit:* Prevent team overload and improve delivery throughput by limiting active work. [Other answers: Focus on idle work, not idle workers. Wait for a complete team.]

Index Number | | | | | | | | |

(d). A retrospective revealed that during sprint execution, progress tracking was inconsistent. Name **two** visual tools that teams can use during sprints to communicate and manage work progress. Using clear examples, explain how each tool contributes to better sprint outcomes (i.e., highlight the impact).

[12 marks]

- *Task Board:* A visual board (physical or digital) that categorizes tasks into columns such as ”To Do”, ”In Progress”, and ”Done”. It provides real-time visibility into the state of each task.
  Example: During a sprint, the team notices that many tasks are stuck in ”In Progress”. This can trigger a discussion in the daily Scrum to reallocate resources and remove bottlenecks.
  Impact: Enhances transparency and accountability, allows quick adjustments during execution, and supports daily inspection and adaptation.

- *Sprint Burndown Chart:* A graph showing the remaining effort (e.g., story points or hours) on the Y-axis versus time (e.g., days of the sprint) on the X-axis. It visually depicts whether the team is on track to complete the planned work.
  Example: On Day 7 of a 10-day sprint, the chart shows a flat line, indicating no recent progress. This insight can trigger a mid-sprint review and reprioritisation of work.
  Impact: Helps teams detect early deviations from the plan, facilitates data-driven discussions, and encourages timely corrective actions.

_____ ********** _____